ISSN No. 0976-5697

Volume 6, No. 5, May - June 2015

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Speculative Block Reuse in GPGPU

Suma S, N.P.Gopalan
Bharathiar university, Coimbatore
NIT, Tiruchirapally

***Abstract*—** Speculative Parallelization is a technique used in the superscalar processors to process and compute the data based on certain predictions like memory, control and data. To reduce memory stalls and to increase the instruction level parallelism, this paper provides a new methodology in using the blocks of GPGPU's to solve many generalized problems as speculative block reuse to store the low or high frequency redundant computations to reuse without executing them repeatedly avoiding repeated execution of the instructions. If the prediction fails, the processor executes normally and it does not alter state of execution. The speculative block reuse method of this paper provides more accuracy in reusing the values from the shared memory.

Keywords— speculative parallelism, block reuse, computations, GPGPU, finite automaton, transition states.

I. INTRODUCTION

During the program execution, some instructions and computations are executed and performed repeatedly with serial execution of programs. These instructions are repeatedly executed and increases the execution of instruction. To reduce instruction level parallelism, the instructions that are executed repeatedly are rearranged /reordered accordingly that these instructions are executed and stored in the memory, whenever the instruction repeats in the execution cycle, the instructions are speculatively fetched from the memory and directly being used for execution without again executes the instructions which avoids usage of many resources and limited resources can be used that in turn increases the performance.

Speculative parallelism is a technology incorporated into present day processors to increase instruction level parallelism and thread level speculations. It is being used in modern microprocessors to enhance the instruction computations performances tremendously. Speculative parallelization technique basically uses many technical aspects such as speculative parallel reuse, precomputation, multithreading and compilation, value reuse such that some part of the program is speculated which is not executed during the program execution. If the predicted result is correct, the values are directly being incorporated into the speculated execution path else the results are squashed if the speculation goes wrong, but the beauty of the execution is that it does not alter the state of the execution of the processor if the prediction id correct, it reduces the instruction cycle, increases instruction execution performance. If the speculation goes wrong, it executes normally, the predicted path is squashed. Hence the state of the processor is unchanged. Many processor architecture adopted this techniques as the value predicted is priory being executed and set with the profiled data, the values are analysed before the prediction being done. So the accuracy of predicting the data as the technique serves to provide the increased performance of data.[2,3]

GPGPU refers to the general purpose graphics processors used in the modern processors for solving general purpose problems along with the huge volume of data .it the hybrid architecture of CPU and GPU together computes the data with coarse grained data by the CPU and fine grained data on GPU and CPU refers to the task parallelism and GPU refers to the data parallelism. In the graphics processors, the architecture consists of the grids. Each of the grid holds blocks and in turn blocks consists of n number of threads. Each thread is identified by the threadID and each block is identified by the blockID. The atomic operations are executed by the each thread in a block using the local memory and registers. The threads of the block cooperate with each other through shared memory. The threads of the block communicate with each other while the blocks cannot communicate with each other. But the blocks can be reused for the computations that occur repeatedly.[4,5]

Jaekyu Lee., Nagesh B. Lakshminarayana., Hyesoon Kim., Richard Vuduc., proposes new hardware and software prefetching mechanisms referred as many thread aware prefetching techniques to GPGPU systems.[6,7]

II. SPECULATIVE BLOCK REUSE

In the graphics processors, the blocks are referred as the specblocks that can be predicted or reused for the computations.The repeatedly used computations or the redundant blocks can be identified and used to increase the performance of instruction precomputation speculatively. The precomputed instructions are unique computations that can either occur as high frequency computations or low frequency computations. The output values are forwarded to speculatively executable instructions, but the threads of the blocks in the graphic processors do not cooperate with each other. The threads use shared memory for synchronization to share data values. The specblocks are determined where the precomputed or reusable values are stored in the reference table. Once the data/value is speculated, the values are referenced to the reference table and used in the computations, the value is committed by the checking for the correctness of the data. The checking of the data values are done through the formal languages as deterministic finite automaton by the compiler.

The steps of the process are as follows.

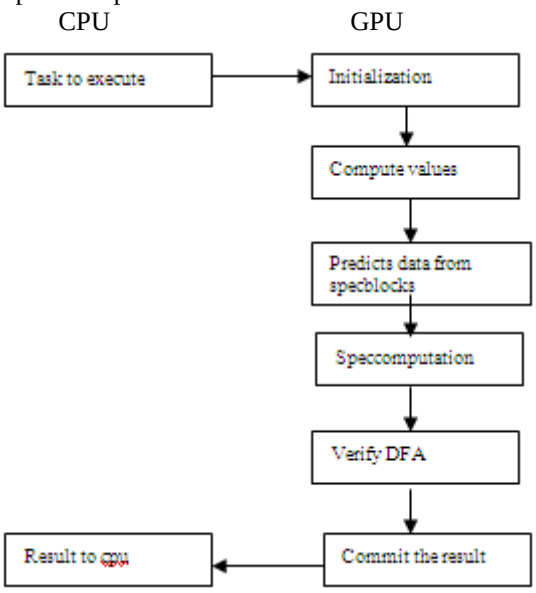

Figure.1 The steps of execution along the CPU&GPU.

Fig. 1 shows the steps of the execution of program on the CPU and GPU.

The transition graph is a finite automaton with the finite number of states and once the input symbol is entered ,it determines the state the machine enters and it may have zero or one transition. On accepting input symbol the machine enters the final state and then the values are committed and sent to the CPU for the result. On rejecting the input symbol by the finite automaton the predicted value/data is wrong and the results are squashed without committing the result and the CPU normally executes the computations and the state of the system is unaltered. The speculated results are correct results in reducing some instructions cycles i.e., fetch and execute cycles such that the performance of the computations or processor is increased.

Consider some general statements like I am latha. I am a girl. In these two sentences, I am repeats with sequential execution of the code, both the statements are executed but with block reuse speculative code, the code is reordered such that I and am are placed in the respective blocks called specblocks and the blockid is referenced in the reference table. When the compiler refers to I and am need not be fetched and executed directly from the shared memory and predicts the results avoiding 2 cycles of memory read operations.

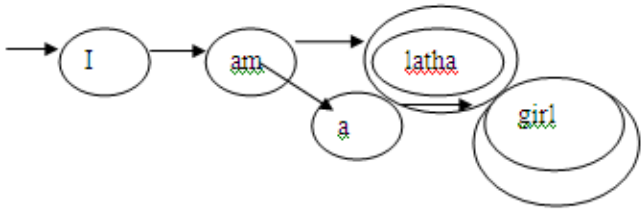

Figure.2 The DFA of committing the input on acceptance.

III.PERFORMANCE

To maximize the performance the execution time should be minimized for any task computer on the machines.

The performance and execution time for any machine is related as

Performance=1/execution time.

CPU execution time for a program=CPU clock cycles for a program X clock cycle time.

The cycle rate and clock cycle time are inverse, so

CPU execution time for a program=CPU clock cycles for a program/ clock rate.[1]

Suppose our program runs in 20 seconds with 400 MHz clock,

The clock cycles required are determined as

CPU time=CPU clock cycles/clock rate

20 seconds=CPU clock cycles/$400*10^6$ cycles/second.

CPU clock cycles=10seconds*$400*10^6$ cycles/second.

CPU clock cycles=$4000*10^6$ cycles.

If the program is reordered and rearranged such CPU clock cycles can be reduces and stalled.

If the same program after reordering takes 6 seconds with speculative block reuse technique, the

CPU clock cycles=6seconds*$400*10^6$ cycles/second.

CPU clock cycles=$2400*10^6$ cycles/second.

So the difference of the CPU clock cycles are

(4000-2400)=1600 cycles/second can be saved.

So for all programs the speculative techniques saves lot of instruction execution cycles/second if they are adopted. Otherwise the CPU execution continues normally without altering the state of the processors. So speculation is not affecting the processor architecture as well as the execution conditions of the system also.

IV.CONCLUSION

As the graphic processors execute the fine grained computations with parallelization parts proved to be good technique with the reusing of the redundant instructions that resulted in saving of the 1600 cycles/second is a comparatively good technique for increasing the instruction execution and reducing the execution time for the computations or the program with parallel techniques.

V. RESULTS

Figure.3 shows the system performance when running a program.

VI. REFERENCES

[1]. David A Patterson., John L. Hennessy., "Computer Organization and Design", Second edition, chapter 2,pg.58-60.

[2]. David Kaeli., pen-chung yew., "Speculative Execution in high performance computer Architectures" Chapman & hall/CRC Chapter13,14.

[3]. Suma S ,N.P. Gopalan," Coalesced Speculative Prefetching and Inter thread Data Dependences IEEE international Conference on Computer Communication and Informatics (ICCCI 2014) Sri Shakthi Engineering college, Coimbatore ,India Jan 3-5 2014. CFP1408R-CDR/ISBN978-1-4799-2352-6/14©2014IEEE.

[4]. NVIDIA. CUDA: Compute Unified Device Architecture. URL http://developer.nvidia.com/cuda.

[5]. NVIDIA. NVIDIA CUDA C Programming Guide version 4.0. 2011.

[6]. Jaekyu Lee., Nagesh B. Lakshminarayana., Hyesoon Kim., Richard Vuduc., "Many-Thread Aware Prefetching Mechanisms for GPGPU Applications" 43rd Annual IEEE/ACM International Symposium on Micro architecture (MICRO), December 2010.

[7]. G. C. Caragea, A. Tzannes, F. Keceli, R. Barua, and U. Vishkin. Resource-aware compiler prefetching for many-cores. In ISPDC-9, 2010.